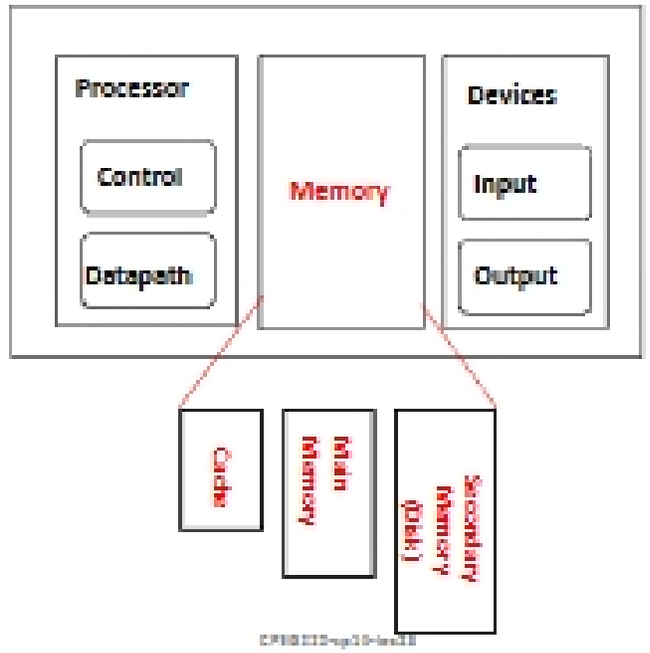
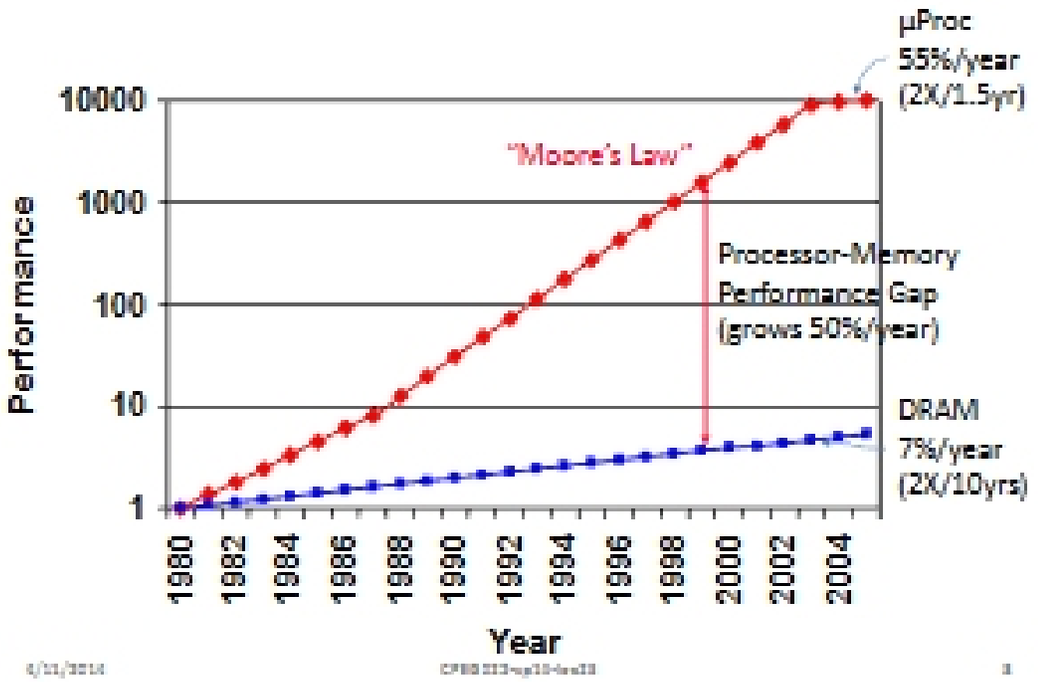
Lecture 23: Memory hierarchy

- Processor-Memory Performance Gap
- Principle of Locality
- Memory Hierarchy

Major Components of a Computer

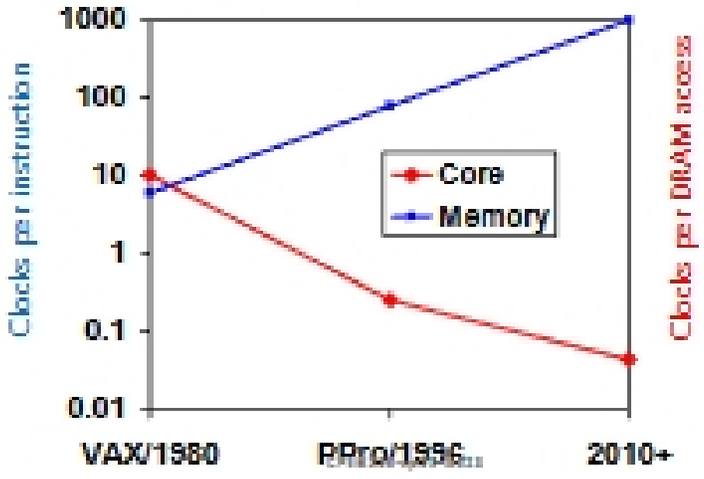


Processor-Memory Performance Gap



The "Memory Wall"

- Processor vs DRAM speed disparity continues to grow
- Good memory hierarchy design is increasingly important to overall performance

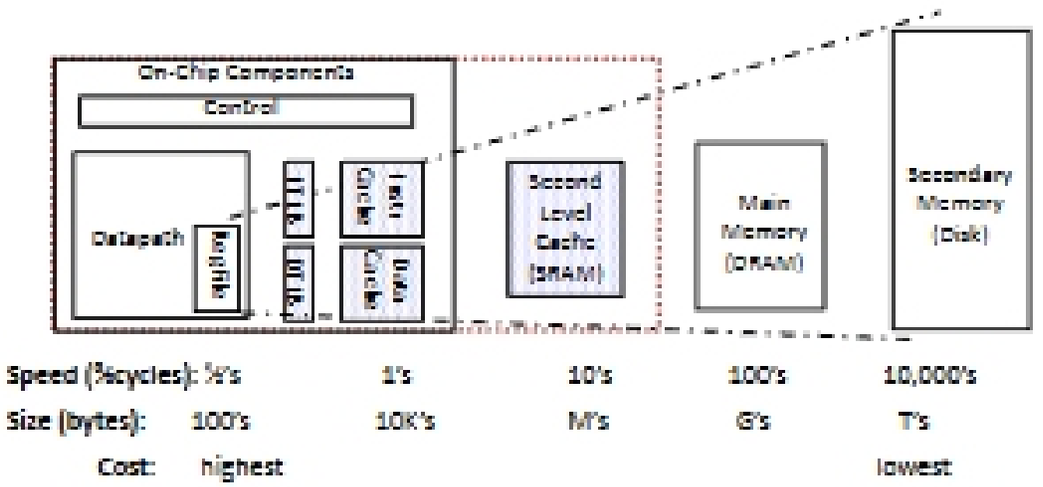


Memory Technology

- Static RAM (SRAM)
  - 0.5ns – 2.5ns, $2000 – $5000 per GB
- Dynamic RAM (DRAM)
  - 50ns – 70ns, $20 – $75 per GB
- Magnetic disk
  - 5ms – 20ms, $0.20 – $2 per GB
- Ideal memory
  - Access time of SRAM
  - Capacity and cost/GB of disk

Large memories are slow and cheap.
Fast memories are small and expensive.

A Typical Memory Hierarchy



Take advantage of the principle of locality

Principle of Locality

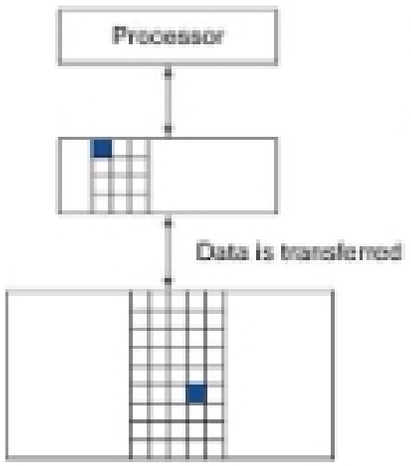
- Programs access a small proportion of their address space at any time
- **Temporal locality**
  - Items accessed recently are likely to be accessed again soon
  - e.g., instructions in a loop, induction variables
- **Spatial locality**
  - Items near those accessed recently are likely to be accessed soon
  - E.g., sequential instruction access, array data

Taking Advantage of Locality

Memory hierarchy:

- Store everything on disk
- Copy recently accessed (and nearby) items from disk to smaller DRAM memory
  - Main memory
- Copy more recently accessed (and nearby) items from DRAM to smaller SRAM memory
  - Cache memory attached to CPU

Memory Hierarchy Levels

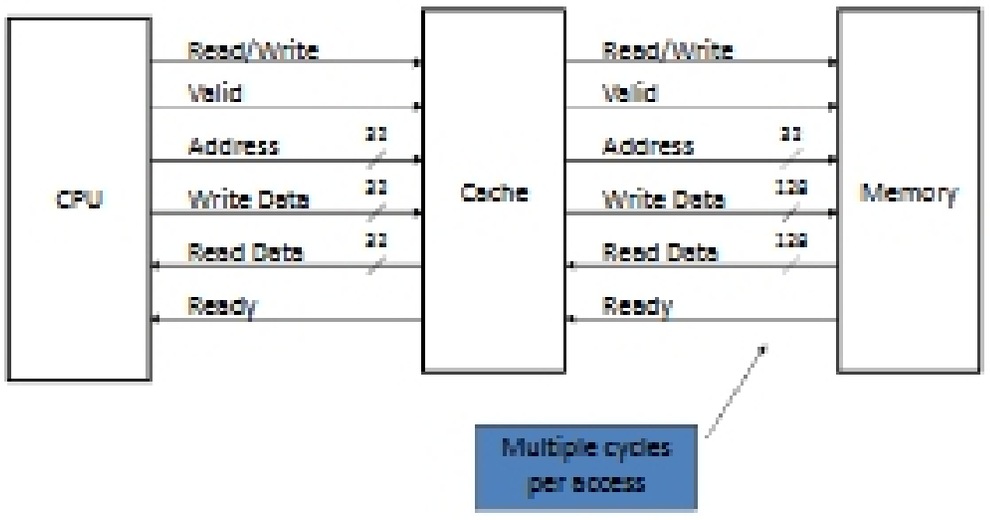


- Block (aka line): unit of copying
  - May be multiple words
- If accessed data is in upper level
  - Hit: access satisfied by upper level
    - Hit ratio: hits/accesses
- If accessed data is absent
  - Miss: block copied from lower level
    - Time taken: miss penalty
    - Miss ratio: misses/accesses
      = 1 – hit ratio
  - Then accessed data supplied from upper level

How is the Hierarchy Managed?

- registers ↔ memory
  - by compiler (programmer?)
- cache ↔ main memory
  - by the cache controller hardware
- main memory ↔ disks
  - by the operating system (virtual memory)
  - virtual to physical address mapping assisted by the hardware (TLB)
  - by the programmer (files)

Interface Signals



Logical (external) memory configuration

* External configurations are tall and narrow
  - More address lines (12 to 20+, typically)
  - Fewer data lines (8 or 16, typically)
* The narrower the configuration
  - The greater the pin efficiency
    - *Adding one address pin cuts data pins in half*
  - The easier the data bus routing
* Many external configurations for given capacity
  - 64 Kb = 64K x 1 (16 A + 1 D = 17 pins)
  - 64 Kb = 32K x 2 (15 A + 2 D = 17 pins)
  - 64 Kb = 16K x 4 (14 A + 4 D = 18 pins)
  - 64 Kb = 8K x 8 (13 A + 8 D = 21 pins)
  - 64 Kb = 4K x 16 (12 A + 16 D = 28 pins)
  - 64 Kb = 2K x 32 (11 A + 32 D = 43 pins)